SZAKDOLGOZAT

Varga Marcell

2014

Pannon Egyetem

Matematika Tanszék

Mérnök informatikus BSc szak

SZAKDOLGOZAT

Képfeldolgozást támogató keretrendszer és modulok készítése

Varga Marcell

Témavezető: Lipovits Ágnes

2014

Ide jön az eredeti vagy a fénymásolt feladatkiírás.

Nyilatkozat

Alulírott Varga Marcell diplomázó hallgató kijelentem, hogy a szakdolgozatot a Pannon Egyetem Matematika Tanszékén készítettem Mérnök informatikus BSc szak (BSc in Computer Engineering) megszerzése érdekében.

Kijelentem, hogy a szakdolgozatban lévő érdemi rész saját munkám eredménye, az érdemi részen kívül csak a hivatkozott forrásokat (szakirodalom, eszközök, stb.) használtam fel.

Tudomásul veszem, hogy a szakdolgozatban foglalt eredményeket a Pannon Egyetem, valamint a feladatot kiíró szervezeti egység saját céljaira szabadon felhasználhatja.

Veszprém, 2014. május 02.

Aláírás

Alulírott Lipovits Ágnes témavezető kijelentem, hogy a szakdolgozatot Varga Marcell a Pannon Egyetem Matematika Tanszékén készítette Mérnök informatikus BSc szak (BSc in Computer Engineering) megszerzése érdekében.

Kijelentem, hogy a szakdolgozat védésre bocsátását engedélyezem.

Veszprém, 2014. május 02.

Aláírás

Köszönetnyilvánítás

Köszönet!

TARTALMI ÖSSZEFOGLALÓ

E szakdolgozat témája ...

Kulcsszavak: *szoftverarchitektúra, képfeldolgozás, adatszerkezetek, adatkezelés, Qt, c++, OpenCV*

ABSTRACT

The topic of this thesis is to ...

Keywords: *software-architecture, image processing, data structure, data processing, Qt, c++, OpenCV*

Todo list

Tartalomjegyzék

1. A feladat összefoglalása

Témám egy olyan képfeldolgozást támogató keretrendszer tervezése és fejlesztése, amely alkalmas képek egyedi vizsgálatára és kötegelt feldolgozására. A feldolgozást végző algoritmusok a dinamikusan betölthető modulokban foglalnak helyet. A rendszer fő haszonélvezője a Pannon Egyetem Képfeldolgozás Kutatólaboratóriuma lesz, de célom, hogy kellően általános rendszer jöjjön létre, amelyet bárki könnyen és egyszerűen használhat, illetve bővítheti saját modulokkal.

1.1. Első lépések

1.1.1. Briefing

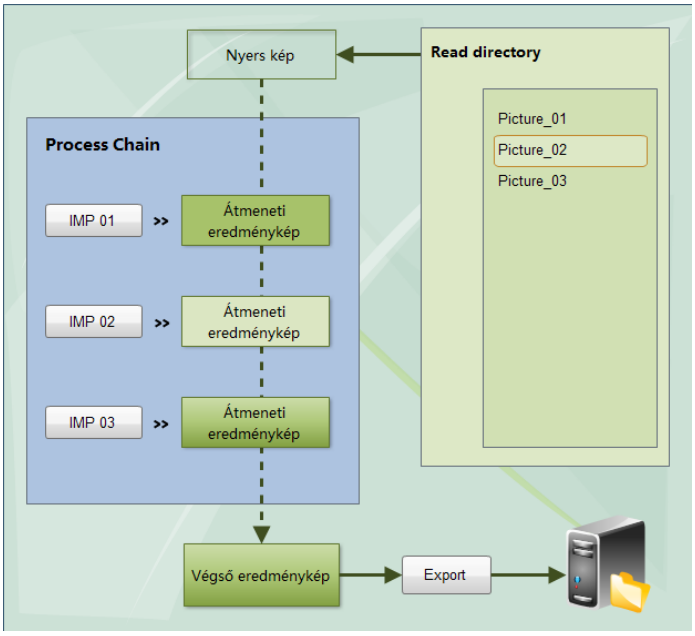
A legtöbb munka során előnyös, ha projekt lényegét megragadva röviden összefoglaljuk a legfontosabb lefutási eset sikeres teljesülését. Erre jó eszköz a rövid (brief[1]) formátumú usecase.
"A felhasználó összeállítja a bemeneti képek, adatforrások listáját. Ezek után meghatározza a feldolgozás lépéseit. Végül megjelöli a kimeneti formát, majd elindítja a feldolgozást. A program egyesével létrehozza a nyers képeket[1]*, majd az adott nyers képen végrehajtja sorrendhelyesen a kijelölt feladatokat, végül a választott kimeneti formába menti ki az eredményképeket*[2]*."*

1.1.2. Egy rövid példa

Az 1. ábrán láthatjuk a rendszer vázlatos működését. Jelen példában egy könyvtárban található összes digitális képet kívánjuk feldolgozni. A beolvasás után a nyers képet átadjuk a feldolgozást végző logikának (Process Chain), ahol jelen példában 3 darab elemi művelet történik (IMP01-03). A feldolgozás utolsó lépése után, a végső eredménykép (jelen esetben) exportálásra kerül, egy a bementi könyvtárral nem megegyező könyvtárba.

[1] Nyersképnek nevezünk minden olyan bemeneti képet, amelyen nem hajtottunk végre semmiféle változtatást.

[2] Átmeneti eredmény képnek nevezzük minden olyan képet, amelyen már végrehajtottunk végbe ment feldolgozás, de nem még nem az összes. Eredmény képnek pedig az olyan képeket nevezzük, amelyeken már lezajlott a feldolgozás



1. ábra. A rendszer vázlatos működése

Fontos megjegyezni, hogy a feldolgozás pipelining [2] jelleget követ, tehát apró elemi lépések sorozata, amelyek kötött sorrendben végzik feladataikat. A műveletekből egy irányított gráf írható fel. Ahol a csúcspontok a műveleti egységek, az irányított élek pedig az adatok áramlása. (Ilyen műveleti módra jó példa különböző grafikus engineknél a fények számítása pl.: [3].)

2. Hasonló célú rendszerek

Következő lépésként megvizsgáltam, hogy milyen hasonló célú szoftverek, illetve szoftvercsomagok találhatóak meg a piacon. Erre azért volt szükség, hogy pontosabb képet kapjak a jelenleg fellelhető megoldásokról, és munkám során az így szerzett pozitív és negatív tapasztalatokat eredményesen hasznosíthassam.

Különböző összehasonlítási szempontokat állítottam fel, melyek lentebb olvashatóak. Az vizsgálat során a személyes benyomáson túl, egyéni véleményeket is figyelembe vettem (pl.: kiadó cégnek vagy alapítványak az ajánlása, vagy független publikáció, újságcikk).

2.1. Összehasonlítási szempontok

2.1.1. Általános tulajdonságok

- Platform: Milyen környezetben és operációs rendszeren használható? Milyen eszközökkel fejlesztették?
  Hordozhatósági szempontonból került be a listára.
- Licence: Milyen licenc alatt került publikálásra?
  Elsősorban pénzügyi és kód újrahasznosíthatósági jellemzők miatt érdekes.
- Cél csoport: A szoftver kinek az igényeinek a kielégítésére törekszik?
  Legtöbb esetben a célcsoport már alapvetően meghatározza, hogy a szoftverbe milyen funkcionalitásokat építünk be, illetve, hogy ezekhez milyen interfészt biztosítunk a jövőbeni felhasználóink részére.
- Támogató: Van hivatalos támogatása? (cég, alapítvány)
  Az esetek jelentős részében megfigyelhető, hogy egy szoftver, szoftvercsomag akkor válik igazán jól támogatottá, ha fejlesztői közösségen kívül egy nagyobb szervezet is gondozásába veszi.
- Felhasználói közösség: Fórum, levelező listák?
  Bármilyen előre nem látható hiba történhet: Ami elromolhat az el is romlik! A fenti csatornákon segítséget kérve nagy eséllyel kaphatunk választ kérdésünkre, és megoldást problémáinkra.
- Plugin rendszer: Plugin betöltésre van lehetőségünk? Saját plugin?
  A képfeldolgozás egy eléggé sokrétű szerteágazó lehetőségeket, funkcionalitásokat magában foglaló szakterület. Így az csak utópisztikus álom, hogy egyszer valaki implementálja az összes funkcionalitást és onnantól kezdve mindenki boldogan használja azokat az idő végezetéig... Ezért ha a program dinamikusan bővíthető (akár a felhasználó által készített bővítményekkel), jelentős előnyt jelent a többi monolitikus rendszerrel szemben.
- Kötegelt feldolgozási lehetőség: Feldolgozhatunk egyszerre nagy mennyiségű képet?
- Automatizálási lehetőségek: Automatizálhatjuk a feldolgozást?
  Ha lehetőségünk van a meglévő egyszerű feldolgozási lépéseket testreszabni,

esetleg összekombinálni akkor az szintén egy jelentős előny lehet, más ilyen lehetőségekkel nem rendelkező szoftverekkel szemben.

- Fejlesztői eszközök: Rendelkezik hivatalos fejlesztői eszközökkel?
- Támogatott bemeneti formátumok köre
- Megjelenítési, vizualizációs lehetőségek listája, módjai
  Hasznos ha több féle interfészt biztosítunk az adott információ megjelenítéshez: más logikai kontextusban helyezve új megfigyeléseket is tehet a felhasználónk.

2.1.2. Képfeldolgozási képességek

- Képjavító eljárások, pl.: élesítés, kontrasztkiegyenlítés
- Geometriai műveletek, pl.: átméretezés, forgatás, tükrözés
- Analizálás, pl.: eltérések detektálása, alacsony szintű képleírók
- Szerkesztési műveletek, pl.: logikai, szöveg, alakzatok elhelyezése
- Színterek közötti konverzió, pl.: RGB $\rightarrow$ HSL, csatornák külön kezelése stb

2.2. Választott szoftverek

- *ImageJ*[4]
  Képfeldolgozást és analizálást végző rendszer, amely a National Institutes of Health fejlesztése. A program első indulásakor látható, hogy itt egy professzionális orvostechnológiai eszközről van szó. Támogatottsága jelentős mind közösségi, mind bővíthetőségi szempontból. Eszközkészletének palettája széleskörű, külön kiemelném Z és T funkciókat.[5]
  A Z funkciók segítségével pl.: MRI-vel készített sorozatos metszeti képeket kezelhetünk könnyedén, lehetőségeinket tovább növeli, hogy a térbeli szervezés mellett még időbeli struktúra felépítésére és kezelésére is lehetőséget ad a program (T funkciók).

- *ImBatch*[6]
  Képfeldolgozást végző rendszer. Célcsoportja egyértelműen egy félprofesszionális felhasználói szint. Tehát itt eleve nem is várunk professzionális analiti-

kai funkciókat. Cserébe kapunk egy szép, letisztult, egyszerű grafikus felhasználói felületet, és egy pár használatot segítő kényelmi funkciót: pl.: Windows helyérzékeny menü intergrációt.

- *OriginLab - Image Processing*[7]
  Az OriginLab szoftver csomag része, amely első sorban tudományos és ipari célközöséget szolgál ki. [8] A korábban tárgyalt rendszerekkel ellentétben ez a szoftver fizetős (21 napos teszt verzió igényelhető). Ára hozza az iparban szokásos szoftver árakat [9], amely személyes felhasználásra kissé borsos, azonban funkcionalitása kárpótolja a felhasználót. Rentgeteg elemzési lehetőség mellett még OriginC-ben saját algoritmusainkat is megvalósíthatunk, a LabView támogatás már majdnem, hogy csak hab a tortán.

2.3. Összefoglalás

2.3.1. Tapasztalatok

A részletes összehasonlítás az 1.4. táblázatból olvasható ki a 19. oldalon.

Az adatsorok elemzése közben, több fontos követelmény, fejlesztési irányvonal körvonalazódott:

- Hordozható legyen több platformra. Hiszen egy laboratóriumban több féle architektúra elő fordul. (Ideális esetben tehát a szoftverünk legyen crossplatform.)
- Nyitott legyen a további fejlesztésekre. A felhasználónak adjuk meg a lehetőséget, hogy testre szabhassa a szoftverünket, vagy akár önmaga is fejlesztővé válhasson, így bővíthesse a pluginek körét, vagy javíthassa a fő programot.
- Szerepeljenek automatizálási lehetőségek. Írhassunk makrókat, vagy vizuális módon szerkeszthessünk algoritmusokat.
- A rendszert ajánlott az adott részterületen leggyakrabban előforduló, és legnépszerűbb ki- és bemeneti adatformátumokra felkészíteni.

- Már gyárilag nagy mennyiségű képjavító, feldolgozó, szerkesztő, elemző és analizáló funkcionalitással érkezzen a szoftver.

2.3.2. Célok

Összegezve láthatjuk, hogy eléggé könnyen lehet már előkészített rendszereket választani a szoftverpiac palettájáról. Ilyenkor jogosan felmerül a kérdés, hogy ez a projekt miben ad többet, mint a jelenlegi lehetőségek?

- Célom, hogy a programba új funkciók integrálása ne ütközzön problémákba, és az ilyen módon implementált funkcióknak a főprogramtól függetlenül is terjesztőknek kell lenniük. Ez természetesen maga után vonzza, hogy a plugin és főprogram közötti interfésznek kellően kompaktnak és univerzálisnak kell lennie, hogy több verzióváltás alatt is változatlan lehessen.

- A sok kis méretű funkció egy idő után kezelhetetlenné válik, ezért legyen lehetőség kategorizálásra.

- Legyen lehetőség átlátható szerkezetű grafikus felhasználói felület használatára. Erre kifejezetten alkalmas a blokkos kapcsolat alapú vizuális szerkesztő. Ez a módszer több különböző technológiai területen sikeresen vizsgázott pl.: Labview blockdiagrammjai [11], vagy UDK4 BluePrint Editorja[12] (amely az UDK3 Kismetjének egy tovább fejlesztett változata).

Licencelési infók

2.4. Hasonló célú rendszerek összehasonlítása táblázat

	ImageJ	ImBatch	OriginLab
Platform	Multi (Java)	Win (C#)	Win (C/C++)
Licence	Public Domain	@TODO	@TODO
Cél csoport	professzionális (orvosi)	félprofesszionális (általános)	professzionális (tudományos, ipari)
Támogató	National Institutes of Health	High Motion Software	OriginLab
Felhasználói közösség	wiki, leírások, fejlesztői dokumtáció, levlista, fórum	gyik, leírások, oktató videók	gyik, wiki, leírások, fejlesztői dokumentáció, fórum
Plugin rendszer	igen	igen	igen
Kötegelt feldolgozás	igen (Z-T funkciók)	igen (akár helyérzékeny menü)	igen
Automatizálás	makrók	részben	originC
Fejlesztői eszközök	igen	igen	igen
Bemeneti formátumok	széleskörű (orvosi irány)	széleskörű (általános irány)	széleskörű (ipari irány)
Megjelenítés, GUI	komplex	egyszerű letisztult	komplex
Képjavító eljárások	igen	igen	igen
Geometriai műveletek	igen	igen	igen
Analizálás	igen	nem	igen
Szerkesztési műveletek	igen	igen	igen
Színterek közötti konverzió	igen	igen	igen

1.1. táblázat. Hasonló célú rendszerek összehasonlítása

3. Rendszertervek

A következőkben röviden összefoglalom a szakdolgozat tárgyát képező szoftver terveinek elő- és elkészítésének menetét. A korábbi fejezet végén vázolni kezdtem a programmal szembeni elvárásokat, javasalatokat, ez gyakorlatilag a követelményrendszer felírásának a kezdeti lépése volt. Természetesen a jelenlegi követelményrendszer kialakítását még megelőzte több beszélgetés, konzultáció is.
A megbeszélések során több logikailag összetartozó objektum, folyamat került felírásra. Ezek rendre nevet kaptak a kommunikáció hatékonyságának növelése érdekében. A legfontosabb elenevezéseket a 1.2 táblázat prezentálja.

3.1. Szószedet

Elnevezés	Deffiníció
Bimg	A főprogramnak az elnevezése (képzése a Batch Image szavakból történt szóösszerántással).
Modul	BIMG-n belüli logikai egység.
Plugin	BIMG dinamikus kiterjesztése, az IMP Nodek lelőhelye
IMP vagy Node	Image Process Node, képfeldolgozási alapegységnek tekínthetjük, egy IMP általában egy jól körülhatárol művelet elvégzésére alkalmas. Három típusa van: adat (DataNode), indikátor (IndicatorNode), és feldolgozó (ProcessNode)
Slot	IMP Node be- és kimeneti interfészei. Egy slot egy paramétert kezel és tárol.
Parameter	Tetszőleges típusú adat. (pl.: szám, szöveg, kép)
Slot Connection	Kettő darab azonos típusú paraméter tartalmazó slot között kapcsolat hozható létre. Ezt a kapcsolatot nevezzük Node Slot Connectionnak, vagy IMP Connection-nak. Kapcsolat csak kimeneti és bemeneti slotok között jöhet létre. Az irány kötelezően: Ki-Be.
Process Chain	Node-kat tartalmaz, és vezérli a feldolgozást.

1.2. táblázat. Szószedet

A továbbiakban ezeket az elnevezéseket használom az adott objektumokra.

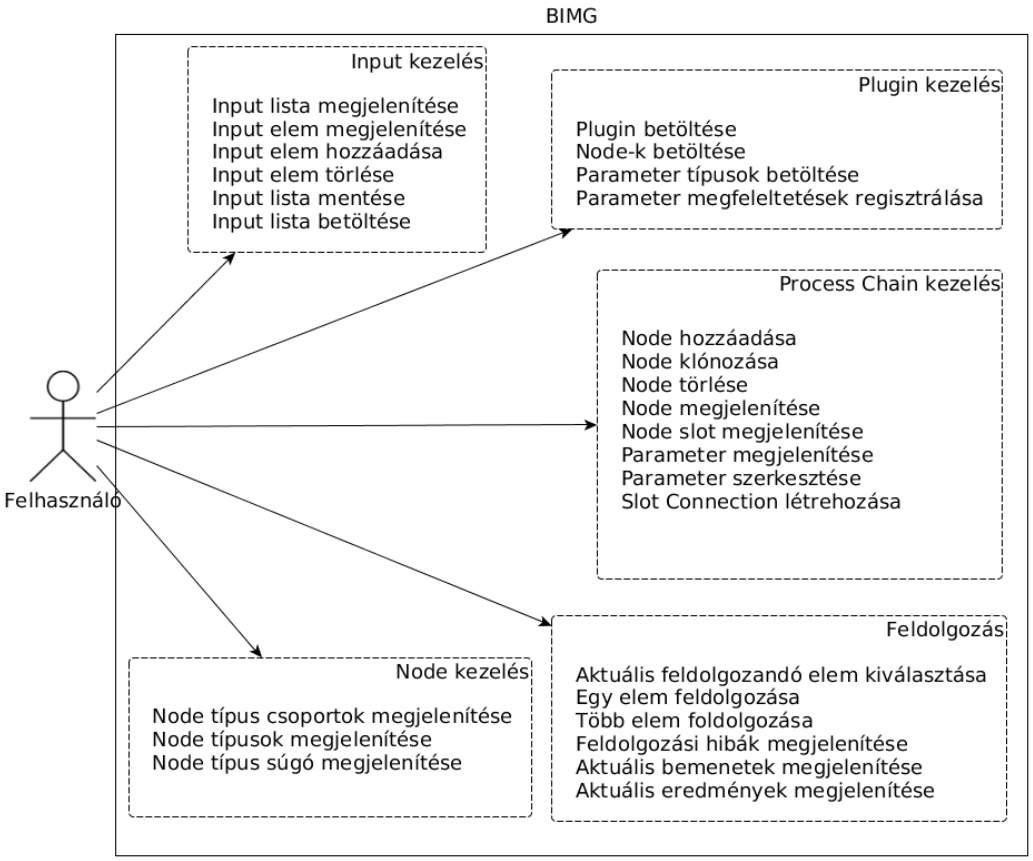
3.2. Követelmény analízis

A követelmény analízist a RUP (Rational Unified Process) metodika FURPS+ rendszerének útmutatása alapján végeztem.
Célom az volt hogy a legtöbb tulajdonság és jellemző feltüntetésre kerüljön amely

szükséges, hogy a szoftver megoldja az adott problémát. [20] Ezek a követelmények jellemzőjüket tekintve lehetnek funkcionális követelmények, és nem funkcionális követelmények.

3.2.1. Funkcionális követelmények

Funkcionális követelményeknek tekintünk minden olyan követelményt, amely a fő termékünkbe valamilyen képességet biztosít. [21] Azaz ide tartozik minden felhasználói, cél feladat és aktivitás. Erre egy kiválló eszköz a usecase, melyet legegyszerűbben a usecasediagramm segítségével tekinthetünk át.
A 2. ábrán megfigyelhető a BIMG sematikus usecase diagrammja[3].



2. ábra. A BIMG sematikus usecase diagrammja

Az ábráról is egyértelműen leolvashatóak rendszert képző legfontosabb logikai egységek:

[3]Megjegyzés: a vázlatos ábrázolás és a könnyebb szemléltetés miatt a felhasználó csak a funkcionalitások csoportjaival és nem a külön-külön a funkcionalitásokkal került összekötésre.

- Input kezelés: a funkcionalitásokból láthatjuk, hogy egy bemeneti elemekből azaz nyers képekből álló struktúrát tudunk managelni.
- Plugin kezelés: itt töldőnek be a feldolgozó egységek és az ezekhez tartozó esetleges, speciális adattagok is. Fontos megjegyezni, hogy szükség van az adattagok közötti konverziós lehetőségek deffiniálására is, hiszen így lesznek képesek a különböző fejlesztésű node-k egymással hatékonyan kommunikálni és együttműködni.
- Node kezelés: a pluginekből beolvasott node-k kezelése törénik ezen a szinten.
- Process Chain kezelés: Ezen a szinten találjuk meg azokat a funkciókat, amelyek a felhasználó számára lehetőséget adnak a feldolgozási folyamat deffiniálásra, szerkesztésére és tesztelésére.
- Feldolgozás: Ezen a szinten történik a feldolgozás összehangolása és az eredmények vizualizálása.

A teljes diagramm a 10.2. ábrán tekínthető meg a 20. oldalon. A kifejtett usecase vázlata a 20 oldaltól kezdődik.

3.2.2. Nem funkcionális követelmények

A nem funkcionális követelmények halmazába tartozik minden, olyan követelmény, amely valamilyen korlátot, megszorítást vagy minőség irányú feltételt deffiniál. [22] Természetesen ide a futás idejű követelményeken túl (mint pl.: megbízhatóság, teljesítmény, hibakezelés), ideértjük a fejlesztési követelményeket is (pl.: modularitás, bővíthetőség, kód újrafelhasználás stb). Először összegyűjtöttem a legfontosabb nem funkcionális követelményeket, majd ezeket osztályoztam, hogy az (F)URPS+ metodika melyik osztályába tartoznak. Ehhez jó kiindulási pont volt a témakiírás és a konzultációs beszélgetések.

3.3. A feladat modellezése - Domain model

A követelmény analízist a megoldandó feladat részletes elemzése és modellezése követte. Erre az egyik legszélesebb körben használt eszköz a domain model. Előnye,

Típus	Követelmény
Modularitás és bővíthetőség	A paraméterezhető képfeldolgozási algoritmusokat dinamikusan betölthető modulok biztosítsák a rendszer számára.
Használhatóság	A könnyen kezelhető grafikus felhasználói felületen legyen mód többlépcsős feldolgozásra.
Támogatás	Amennyiben a felszhasználói felület használata nem teljesen triviális, készüljön hozzá kezelési leírás.
Teljesítmény és megbízhatóság	A rendszer legyen képes nagy mennyiségű bemenet feldolgozására.
Megbízhatóság	Egy esetleges hibás működésű modul ne okozzon rendszer szintű problémát.
Implementáció	A fejlesztés lehetőleg modern, ingyenes, nyíltforrású technológiákkal történjen.
Interfész	Több ki és bemeneti formátum támogatása.

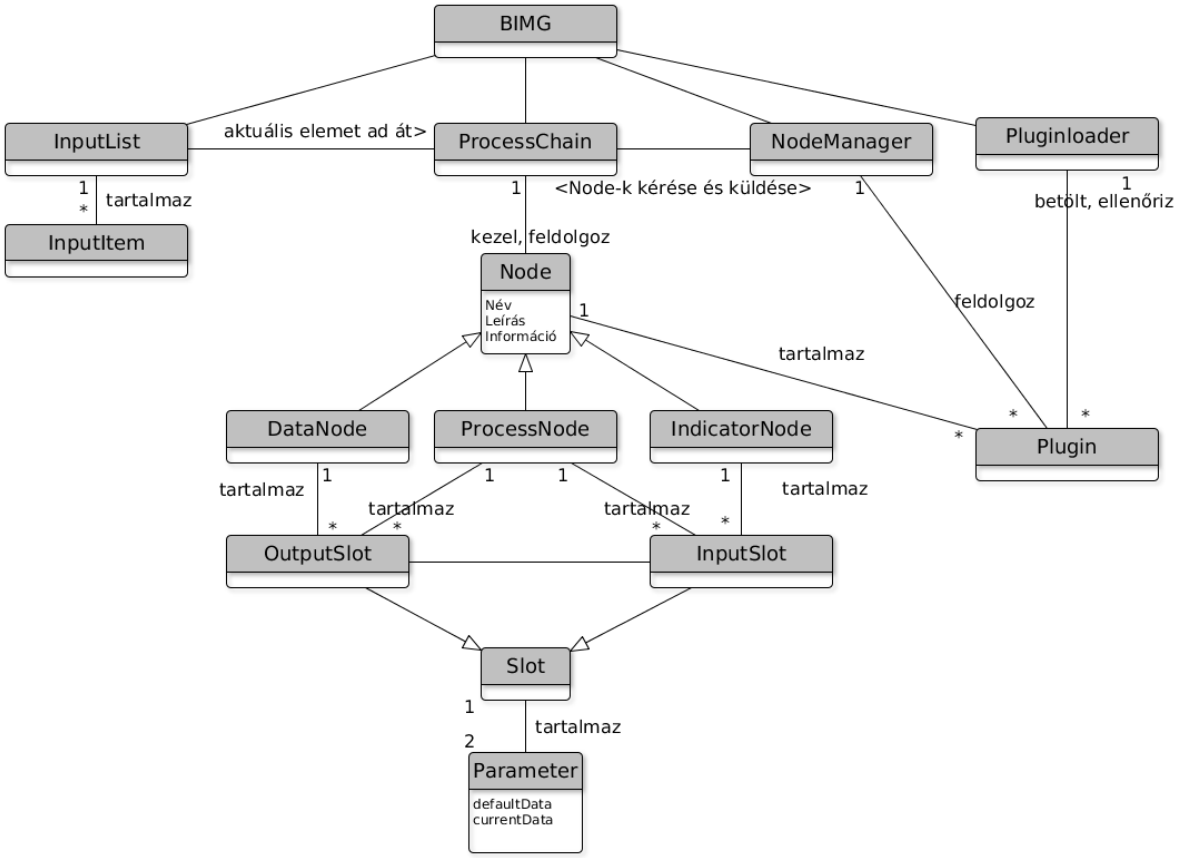
1.3. táblázat. A legfontosabb nem funkcionális követelmények

hogy kevés objektummal egyszerűen, és érthetően tudjuk ábrázolni a megoldani kívánt problémakört és feladatokat.[1]

A problémát bemutató domain modell a 3. ábrán látható. Itt szeretném megjegyezni, hogy ez a model nem az egész rendszer ábrázolja. Itt csak alaprendszerek vázlatos bemutatására szorítkozom: Node-Slot-Parameter, Process Chain, Input, Plugin, Node Management. A teljes modell a 4. ábrán az 35. oldalon látható. A többi alrendszerről pedig a későbbiekben lesz szó, hiszen azok, már jelentős mértékben függenek a választott technológiáktól, és nem képzik szerves részét az alap logikának.

3.3.1. Az alap rendszer

A BIMG alapját a Node-Slot-Parameter rendszer képezi. Paraméter lehet bármilyen tetszőleges adat (szám, szöveg, kép, vektor, mátrix stb).
Minden slotnak kötelezően két paraméterrel kell rendelkeznie, amíg az első az alapértelmezett értéket, addig a második az éppen aktuális értéket reprezentálja. A két érték típusa mindig azonos. Így látható, hogy a slotok csoportosíthatóak tárolt adattípus szempontjából, azonban ezen túlmenően működési irány alapján is rendsze-



3. ábra. A probléma ábrázolása domain modellen.

rezhetőek. Az irány azt reprezentálja, hogy az adott slot az őt tartalmazó node-ban milyen irányú kommunikációt képes végezni.

- InputSlot: Bemeneti kommunikációt végez, így bemeneti adatot reprezentál a node szempontjából. Alapértelmezetten blokkolja a node végrehajtását.
- OutputSlot: A node szempontjából kimeneti adatot reprezentál, hiszen kimeneti kommunikációt végez. A node végrehajtását nem blokkolja.

Összegezve beszélhetünk akár pl.: bemeneti számot, kimeneti képet stb kezelő slotokról. Egy slot csak egy nodehoz tartozhat, de egy node több slotot is foglalhat magában. Három féle node létezik a rendszerben:

- DataNode: Csak Output slottal rendelkezik, tehát a teljes feldolgozás szempontjából egyértelműen csak bemeneti adatforrásnak tekinthető. Mivel nincsen bemeneti slotja azonnal végrehajtható. Alapvető funkciója: adatot juttat be a ProcessChainbe.

- IndicatorNode: csak Input slottal rendelkezik, tehát a teljes feldolgozás szempontjából csak kimeneneti adatforrásként kezelhető. Alapvető funkciója: adatot juttat ki a ProcessChainból.
- ProcessNode: Input- és OutputSlotokkal is rendelkezik. Az egyetlen node típus, amely igazi feldolgozási logikával rendelkezik (tehát nem csak megjelenít, vagy vissza adértékeket hanem azokkal tényleges műveletet is végez). Végrehajtása az InputSlotok miatt alap esetben blokkolt. Alapvető funkciója: feldolgozás.

"Node kapcsolat info"

A nodek között slotok segítségével kapcsolat építhető fel. Bővebben: A nodeknak a végrehajtását a ProcessChain ütemezi és irányítja. Az éppen feldolgozásra szánt kép az InputListről érkezik, amelyet a ProcessChain átvesz. Az InputListben több elem is található, mindegyik egy nyers eredményképet reprezentál.
A nodek plugienkben kerülnek a rendszerbe. A plugineket a pluginmanager tölti be induláskor automatikusan, majd validálás után a plugin nyers tartalmát tovább adja a NodeManagernek. A NodeManger az átvett nyers plugin tartalmat, amely, különböző BIMG által használt objektumok listái, feldolgozza, és regisztrálja az adott node típust. Amennyiben a felhasználó a ProcessChaint egy új noddal szeretné bővíteni, a ProcessChain kérést intéz a NodeManagerhez, ami ha a kért node típus érvényes és regisztrált elkészít egy példányt, és visszadja a ProcessChain számára.

3.4. Választott technológiák

A feladat modellezése után döntést kellett hoznom, hogy milyen technológiákkal kívánom megvalósítani a tervezett rendszert. Több tényező is befolyásolta a döntésemet. Ezekeből két nagy csoportot írtam fel: a feladatból illetve szubjektív nézőpontból kiemelt fontosságú szempontok. A feladatból, és követelményrendszerből adódóakat a korábbi fejezetekben már részleteztem, ezért a következőkben csak a szubjektív pontokat vázolnám fel.

- Egyszerű és gyors, minőségi fejlesztés
- Könnyű dokumentálhatóság
- Legyen korábbi munkáimból rutinom az adott technológiák alkalmazásában

- Képfeldolgozási függvénykönyvtárakkal legyenek jól ellátottak a kiválasztott technológiák (nem szeretném újra feltalálni a kereket)

A program alap szerkezete Qt-val, a képfeldolgozásért felelős komponensek OpenCV-vel történő implementálása mellett döntöttem. A verziókövetést Git-el végeztem, a dokumentáció és dolgozat elkészítéséhez pedig Latex-et, Gummi-t, Doxygen-t, és Yed-et használtam.

3.4.1. Qt

Egyike a legmeghatározóbb[19] multiplatform c++-ra épülő alkalmazás keretrendszerenek. [13] Korábban már több másik projektben is sikeresen dolgoztam vele. A részletes dokumentáció és aktív felhasználói/fejlesztői bázis sokat segített a fejlesztésben. [14] [15][16] Licencelése kedvező, elérhető OpenSource és Enterprise verziója is. Olyan nagy cégek is használják mint a BlackBerry, Michelin vagy a Panasonics.[17]
A fejlesztés korai fázisa az 5.1es verzióval történt, azonban az új 5.2es verzió jelentős újításokat hozott (főként a meta-type rendszer terén történő változások hatottal a projektre), ezért átálltam az 5.2.1-es verzióra.

3.4.2. OpenCV

Open Source Computer Vision Library, nyíltforrású képfeldolgozást és gépi tanulást megvalósító függvénykönyvtár.[23] Natív c++-ban implementált és erősen támaszkodik azt stl tárolókra. Sajnos gyárilag csak c/c++ és phyton-al tud hatékonyan együttműködni. Szerencsére egy vékony wrapper elkészítésével könnyen összekapcsolható Qt-val is. Funkcionalitásával széleskörű feladatok megoldására kiválóan alkalmas, technológiai lehetőségek arzenálját vonultatja fel pl.: Cuda, OpenCl. Multipaltform, még okostelefonra is elérhető a portja (Android 2010, iOS 2012). Dokumentációja is megfelelő. Jelen programban a 2.4.8-as verzióval dolgoztam.

4. Architektúrális tervek

.

5. Fejlesztési napló

.

6. A Fejlesztés részletei

.

7. Az elkészült munka értékelése

.

8. Továbbfejlesztési lehetőségek

.

9. Irodalomjegyzék

[1] Craig Larman (2004). *Applying UML and Patterns, Prentice Hall, 3 edition 6.7 (66)(127)*

[2] Jack J. Dongarra (1995). *Numerical Linear Algebra on High-Performance Computers (3)*

[3] http://www.valvesoftware.com/publications/2006/ SIGGRAPH06_Course_ShadingInValvesSourceEngine_Slides.pdf *Valve, Jason Mitchell (2007), Shading in Valve's Source Engine (37)*

[4] http://imagej.nih.gov/ij/ *ImageJ - Image Process and Analysis in Java*

[5] Tony J. Collinsm, *ImageJ for microscopy* BioTechniques 43:S25-S30 (July 2007)

[6] http://www.highmotionsoftware.com/products/imbatch *ImBatch - Batch Image Processing Software*

[7] http://www.originlab.com/index.aspx?go=Products/Origin/DataAnalysis/ImageProcessing *OriginLab - Image Processing*

[8] http://www.originlab.com/index.aspx?go=COMPANY/AboutUs *OriginLab - About Us*

[9] http://www.originlab.hu/Originv9_USD_NEW_20121022_WEB.pdf
OriginLab - Licences

[10] http://www.ibm.com/developerworks/rational/library/4706.html
IBM - Capturing Architectural Requirements

[11] http://zone.ni.com/reference/en-XX/help/371361J-01/lvconcepts/blockdiagram/
NI - LabVIEW 2012 Help - Block Diagram

[12] https://docs.unrealengine.com/latest/INT/Engine/Blueprints/Editor/index.html
UDK4 - Blueprint Editor Reference

[13] http://qt.digia.com/About-Us/ *QT - About*

[14] http://qt-project.org/doc/ *QT - Doc*

[15] http://qt-project.org/forums *QT - Forums*

[16] http://lists.qt-project.org/mailman/listinfo *QT - MailingLists*

[17] http://qt.digia.com/Qt-in-Use/ *QT Digia - In Use*

[18] http://opencv.org/about.html *OpenCV - About*

[19] http://blog.qt.digia.com/blog/2014/04/16/qt-5-2-over-1-million-downloads/
QT Digia - Over 1 million

[20] http://www.math.unipd.it/ tullio/IS-1/2007/Approfondimenti/SWEBOK.pdf
Guide to the Software Engineering Body of Knowledge, Chapter 2 - SOFTWARE R EQUIREMENTS

[21] http://www.ibm.com/developerworks/rational/library/4706.html#N10098
IBM - Capturing Architectural Requirements, Functional Requirements

[22] Ruth Malan and Dana Bredemeyer http://www.bredemeyer.com/pdf_files/NonFunctReq.PDF
Architecture Resources For Enterprise Advantage (2)

[23] http://opencv.org/about.html *OpenCV - About*

10. Mellékletek

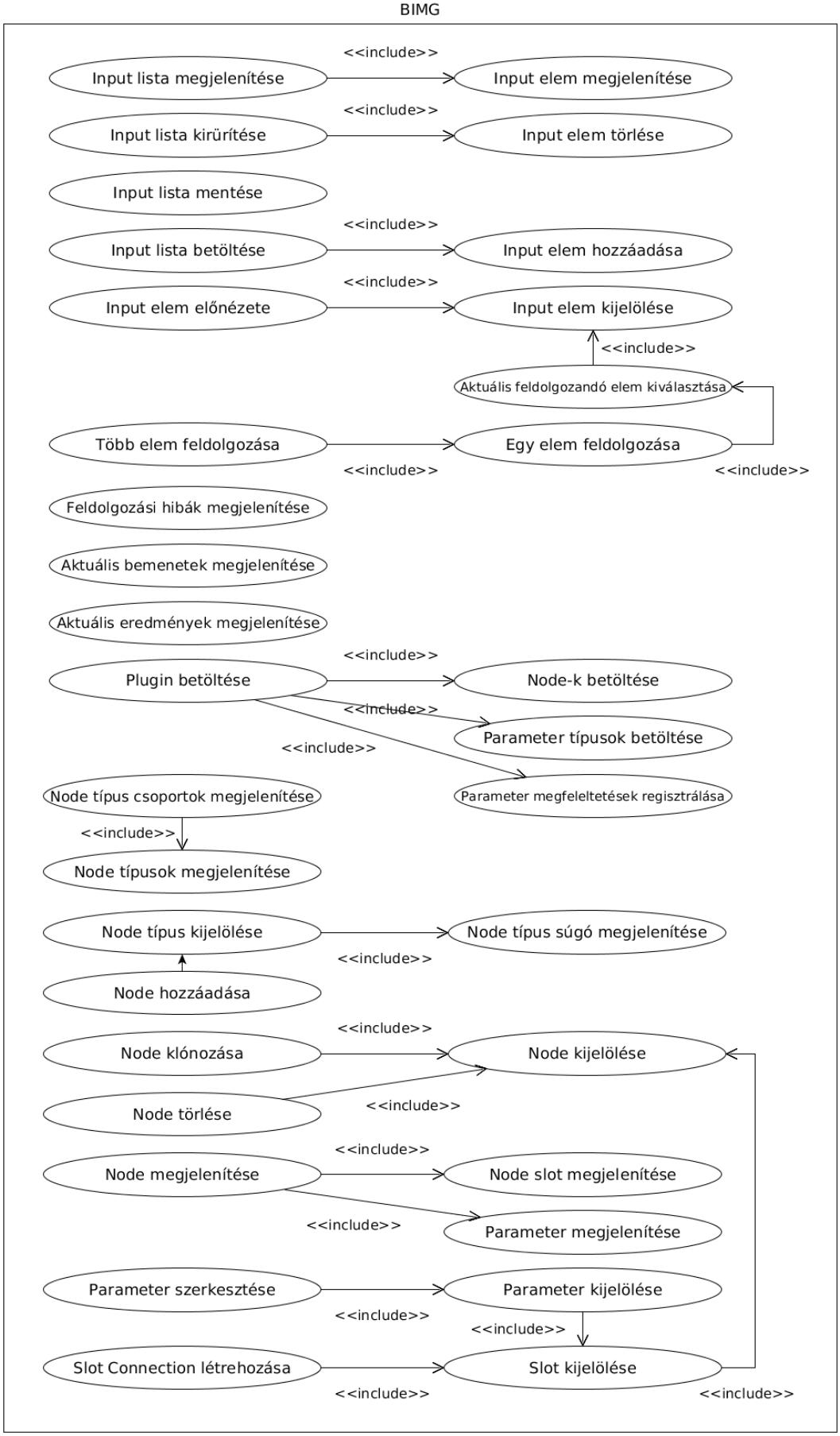
Licencelési infók

10.1. Hasonló célú rendszerek összehasonlítása táblázat

	ImageJ	ImBatch	OriginLab
Platform	Multi (Java)	Win (C#)	Win (C/C++)
Licence	Public Domain	@TODO	@TODO
Cél csoport	professzionális (orvosi)	félprofesszionális (általános)	professzionális (tudományos, ipari)
Támogató	National Institutes of Health	High Motion Software	OriginLab
Felhasználói közösség	wiki, leírások, fejlesztői dokumtáció, levlista, fórum	gyik, leírások, oktató videók	gyik, wiki, leírások, fejlesztői dokumentáció, fórum
Plugin rendszer	igen	igen	igen
Kötegelt feldolgozás	igen (Z-T funkciók)	igen (akár helyérzékeny menü)	igen
Automatizálás	makrók	részben	originC
Fejlesztői eszközök	igen	igen	igen
Bemeneti formátumok	széleskörű (orvosi irány)	széleskörű (általános irány)	széleskörű (ipari irány)
Megjelenítés, GUI	komplex	egyszerű letisztult	komplex
Képjavító eljárások	igen	igen	igen
Geometriai műveletek	igen	igen	igen
Analizálás	igen	nem	igen
Szerkesztési műveletek	igen	igen	igen
Színterek közötti konverzió	igen	igen	igen

1.4. táblázat. Hasonló célú rendszerek összehasonlítása

10.2. Teljes usecase vázlat


BIMG
Input lista megjelenítése
<<include>>
Input elem megjelenítése
Input lista kirürítése
<<include>>
Input elem törlése
Input lista mentése
Input lista betöltése
<<include>>
Input elem hozzáadása
Input elem előnézete
<<include>>
Input elem kijelölése
<<include>>
Aktuális feldolgozandó elem kiválasztása
Több elem feldolgozása
<<include>>
Egy elem feldolgozása
<<include>>
Feldolgozási hibák megjelenítése
Aktuális bemenetek megjelenítése
Aktuális eredmények megjelenítése
Plugin betöltése
<<include>>
Node-k betöltése
<<include>>
Parameter típusok betöltése
<<include>>
Parameter megfeleltetések regisztrálása
Node típus csoportok megjelenítése
<<include>>
Node típusok megjelenítése
Node típus kijelölése
<<include>>
Node típus súgó megjelenítése
Node hozzáadása
Node klónozása
<<include>>
Node kijelölése
Node törlése
<<include>>
Node megjelenítése
<<include>>
Node slot megjelenítése
<<include>>
Parameter megjelenítése
Parameter szerkesztése
<<include>>
Parameter kijelölése
<<include>>
Slot Connection létrehozása
<<include>>
Slot kijelölése
<<include>>

Használati eset: Input lista megjelenítése
Elsődleges aktor: Felhasználó
Szereplők és érdekeik:
Felhasználó – az aktuális input lista megjelenítése
Elő feltételek:
- Létezzen egy inputlista

Sikeres végrehajtás (utófeltételek):
- Az aktuális lista formázva megjelenítésre került a felhasználó számára

Fő (sikeres) szcenárió:
1. A felhasználó az input lista megjelenítését kezdeményezi
2. A rendszer az input lista elemeit egyesével megjeleníti <<include>> Input elem megjelenítése

Alternatív szcenáriók:
2a. A listában nem találhatóak elemek
1. Üres lista megjelenítése

Használati eset: Input elem megjelenítése
Elsődleges aktor: Felhasználó
Szereplők és érdekeik:
Felhasználó – egy input elem megjelenítése
Elő feltételek:
- Létezzen egy input elem

Sikeres végrehajtás (utófeltételek):
- Az input elem megjelenítésre került a felhasználó számára

Fő (sikeres) szcenárió:
1. Input elem adattagjainak megjelenítése (adattagok kiegészítő specifikációban részletezve)

Alternatív szcenáriók:

Használati eset: Input lista kiürítése
Elsődleges aktor: Felhasználó
Szereplők és érdekeik:
Felhasználó – egy input lista összes elemének törlése
Elő feltételek:
- Létezzen egy kiürítésere váró input lista

Sikeres végrehajtás (utófeltételek):

- Az input lista üres

Fő (sikeres) szcenárió:

1. A felhasználó az input lista kiürítését kezdeményezi
2. A rendszer az input lista elemeit egyesével törli <<include>> Input elem törlése

Alternatív szcenáriók:

2a. A listában nem találhatóak elemek

1. A kiürítési művelet automatikusan sikeres volt

Használati eset: Input elem törlése
Elsődleges aktor: Felhasználó
Szereplők és érdekeik:

Felhasználó – egy input elem megjelenítése

Elő feltételek:

- Létezzen egy input elem

Sikeres végrehajtás (utófeltételek):

- Az input elem törlésre kerül

Fő (sikeres) szcenárió:

1. Input elem eltávolítása a listájáról
2. Input elem és adat tagjainak törlése

Alternatív szcenáriók:

1a. Az input elem nem része semmi féle listának

Használati eset: Input lista mentése
Elsődleges aktor: Felhasználó
Szereplők és érdekeik:

Felhasználó – egy input lista mentése

Elő feltételek:

- Létezzen egy input lista

Sikeres végrehajtás (utófeltételek):

- Az input lista mentése sikeresen megtörtént

Fő (sikeres) szcenárió:

1. A felhasználó kezdeményezi az input lista mentését
2. A rendszer információt kér a felhasználótól a mentés helyére vonatkozólag
3. A rendszer egyesével menti az input lista elemeit: <<include>> Input elem mentése
4. A rendszer visszajelzést ad a felhasználónak a mentés sikerességéről

Alternatív szcenáriók:

2a: Nem valós mentési hely

1. A rendszer visszajelzést ad a felhasználónak a hibáról

2b: Nem elérhető mentési hely
1. A rendszer visszajelzést ad a felhasználónak a hibáról
3a: Nem található elem az input listában
1. A rendszer egy üres listát hoz létre

Használati eset: Input elem mentése
Elsődleges aktor: Felhasználó
Szereplők és érdekeik:
Felhasználó – egy input elem mentése
Elő feltételek:
- Létezzen egy input elem

Sikeres végrehajtás (utófeltételek):
- Az input elem mentésre került

Fő (sikeres) szcenárió:
1. Input elem elérési útjának mentése

Alternatív szcenáriók:

Használati eset: Input lista betöltése
Elsődleges aktor: Felhasználó
Szereplők és érdekeik:
Felhasználó – egy korábban mentett inputlista betöltése
Elő feltételek:
- Létezzen egy input lista

Sikeres végrehajtás (utófeltételek):
- Az input lista összes eleme betöltésre került

Fő (sikeres) szcenárió:
1. A rendszer információt kér a felhasználótól a lista elérési helyére vonatkozólag
2. A rendszer beolvassa a megadott leírót
3. A leíróban található elemekből input elemet épít: <<include>>Input elem hozzáadása

Alternatív szcenáriók:
1a: Nem valós mentési hely
1. A rendszer visszajelzést ad a felhasználónak a hibáról
1b: Nem elérhető mentési hely
1. A rendszer visszajelzést ad a felhasználónak a hibáról
3a: Nem található elem az input listában
1. A rendszer egy üres listát hoz létre

Használati eset: Input elem hozzáadása
Elsődleges aktor: Felhasználó
Szereplők és érdekeik:
Felhasználó – egy input elem hozzáadása az aktuális inputlistához
Elő feltételek:
- Létezzen egy inputlista
- Létezzen egy input elem

Sikeres végrehajtás (utófeltételek):
- Az input elem hozzá lett adva az input listához

Fő (sikeres) szcenárió:
1. Input elem adattagjainak ellenőrzése és felépítése (adattagok kiegészítő specifikációban részletezve)
2. Input elem hozzáadása a listához

Alternatív szcenáriók:

Használati eset: Input elem előnézete
Elsődleges aktor: Felhasználó
Szereplők és érdekeik:
Felhasználó – egy input elem előnézete
Elő feltételek:
- Létezzen egy input elem

Sikeres végrehajtás (utófeltételek):
- Az input elem előnézete megjelenítésre került a felhasználó számára

Fő (sikeres) szcenárió:
1. Input elem képiinformációjának megjelenítése a felhasználó számára előzetes vizsgálat céljából

Alternatív szcenáriók:
1a: Az input elem képiinformációja sérült, vagy nem elérhető
1. Üres/NULL kép megjelenítése

Használati eset: Input elem kijelölése
Elsődleges aktor: Felhasználó
Szereplők és érdekeik:
Felhasználó – egy input elem kijelölése
Elő feltételek:
- Létezzen egy input elem az aktuális listában

Sikeres végrehajtás (utófeltételek):

- Az input elem legyen kijelölve, hogy a felhasználó további műveleteket végezhessen vele

Fő (sikeres) szcenárió:

1. Az input elem kijelölése megtörtént

Alternatív szcenáriók:

Használati eset: Aktuális feldolgozandó elem kiválasztása
Elsődleges aktor: Felhasználó
Szereplők és érdekeik:

Felhasználó – egy input elem kijelölése a esetleges későbbi egy elem feldolgozásra

Elő feltételek:

- Létezzen kijelölt elem: <<include>>Input elem kijelölése
- Létezzen egy process chain

Sikeres végrehajtás (utófeltételek):

- Az input elem kijelölésre került egy elem feldolgozásra

Fő (sikeres) szcenárió:

1. <<include>> Input elem kijelölése
2. ProcessChain default értékekkel rendelkező ImgDataNode-k benépesítése az input elemmel <<include>> Aktuális bemenetek megjelenítése

Alternatív szcenáriók:

Használati eset: Egy elem feldolgozása
Elsődleges aktor: Felhasználó
Szereplők és érdekeik:

Felhasználó – egy elem feldolgozása sikeresen megtörténjen

Elő feltételek:

- Legyen egy kiválasztott feldolgozandó elem: <<include>> Aktuális feldolgozandó elem kiválasztása

Sikeres végrehajtás (utófeltételek):

- A kiválasztott elem feldolgozása sikeresen megtörtént

Fő (sikeres) szcenárió:

1. Process Chain feldolgozási logikájának futtatása (részletek a kiegészítő specifikációban)
2. <<include>> Aktuális eredmények megjelenítése

Alternatív szcenáriók:

1a: Feldolgozási hiba történt futtatás közben

1. Hiba megjelenítése
2. Feldolgozás megállítása

Használati eset: Több elem feldolgozása
Elsődleges aktor: Felhasználó
Szereplők és érdekeik:
Felhasználó – az input lista összes eleme feldolgozásra kerüljön
Elő feltételek:
- Létezzen egy input lista

Sikeres végrehajtás (utófeltételek):
- Az összes elem feldolgozása sikeresen megtörtént

Fő (sikeres) szcenárió:
1. Az input lista elemeinekre egyesével végrehajtjuk az Egy elem feldolgozása <<include>>

Alternatív szcenáriók:
1a: Valamelyik elem feldolgozásakor hibatörténet
1. Feldolgozás megállítása (további elemek nem kerülnek feldolgozásra)

Használati eset: Aktuális bemenetek megjelenítése
Elsődleges aktor: Felhasználó
Szereplők és érdekeik:
Felhasználó – egy bállított input elem megjelenítése a process chain bemeneti ImgDataNode-jain
Elő feltételek:
- Létezzen egy input elem
- Létezzen egy process chain

Sikeres végrehajtás (utófeltételek):
- Az input elem bemásolásra került az összes input elem részére

Fő (sikeres) szcenárió:
1. Megfelelő imgDataNode-k adatainak módosítása az input elem tartalmáva

Alternatív szcenáriók:
1a. Nincsennek benépesítendő ImgDataNode-k
1. Nincsen érdemi akció: nem történik művelet

Használati eset: Aktuális eredmények megjelenítése
Elsődleges aktor: Felhasználó
Szereplők és érdekeik:
Felhasználó – a kimeneti eredményképek megjelenítése a process chain megfelelő ImgIndicatorNode-jain sikeresen megtörtént
Elő feltételek:
- Létezzen egy input elem
- Létezzen egy process chain ahol a feldolgozás sikeres volt

Sikeres végrehajtás (utófeltételek):

- Az eredmény kép vizualizálásra került a felhasználó számára

Fő (sikeres) szcenárió:

1. Eredménykép vizualizálása a felhasználó számára

Alternatív szcenáriók:

Használati eset: Plugin betöltése
Elsődleges aktor: Felhasználó
Szereplők és érdekeik:

Felhasználó – a rendszerbe bekerült a plugin

Elő feltételek:

- A felhasználó kezdeményezze a rendszer indulását

Sikeres végrehajtás (utófeltételek):

- Az összes plugin betöltésre került a rendszerbe

Fő (sikeres) szcenárió:

1. Munka könyvtár pharsolása plugin-okra (dll,so)
2. Plugin betöltése
3. Pluginterfész ellenőrzése
4. <<include>>Nodek betöltése
5. <<include>>Parameter típusok betöltése
6. <<include>> Parameter megfeleltetések regisztrálása

Használati eset: Node-k betöltése
Elsődleges aktor: Felhasználó
Szereplők és érdekeik:

Felhasználó – Node-k használatra készek

Elő feltételek:

- Valid plugin legyen a rendszerben
- Node leíró kezelő adatok fogadására készen álljon

Sikeres végrehajtás (utófeltételek):

- Node-k használatra készek

Fő (sikeres) szcenárió:

1. A pluginból kikéri a rendszer az elérhető pluginokat
2. A Node ellenőrzésre kerül
3. A Node leírója bemásolásra kerül a node leírók közé

Alternatív szcenáriók:

1a: Nincs plugin : nincs művelet
2a: Ellenőrzésen elbukik a node: nincs művelet, betöltése a nodenak megszakad

---

Használati eset: Parameter típusok betöltése
Elsődleges aktor: Felhasználó
Szereplők és érdekeik:
Felhasználó – Parameterek használatra készek
Elő feltételek:
- Valid plugin legyen a rendszerben
- Parameterleíró kezelő adatok fogadására készen álljon

Sikeres végrehajtás (utófeltételek):
- Parameter-ek használatra készek

Fő (sikeres) szcenárió:
1. A pluginból kikéri a rendszer az elérhető parametereket
2. A parameter ellenőrzésre kerül
3. A parameter leírója bemásolásra kerül a parameter leírók közé

Alternatív szcenáriók:
1a: Nincs plugin : nincs művelet
2a: Ellenőrzésen elbukik a parameter: nincs művelet, betöltése a parameternek megszakad

---

Használati eset: Parameter megfeleltetések regisztrálása
Elsődleges aktor: Felhasználó
Szereplők és érdekeik:
Felhasználó – Parameter megfeleltetés készek
Elő feltételek:
- Valid plugin legyen a rendszerben
- Parameter megfeleltetés leíró kezelő adatok fogadására készen álljon

Sikeres végrehajtás (utófeltételek):
- Parameter megfeleltetés-ek használatra készek

Fő (sikeres) szcenárió:
1. A pluginból kikéri a rendszer az elérhető parameter megfeleltetéseket
2. A parameter megfeleltetés ellenőrzésre kerül
3. A parameter megfeleltetés leírója bemásolásra kerül a parameter megfeleltetés leírók közé

Alternatív szcenáriók:
1a: Nincs plugin : nincs művelet
2a: Ellenőrzésen elbukik a parameter megfeleltetés: nincs művelet, betöltése a parameter megfeleltetésnek

---

Használati eset: Node típus csoportok megjelenítése

Elsődleges aktor: Felhasználó
Szereplők és érdekeik:
Felhasználó – Node típus csoportok megjelenítése történtjen meg
Elő feltételek:
- Legyenek betöltött nodek és node típusok a rendszerben

Sikeres végrehajtás (utófeltételek):
- A node típus csoportok megjelenítésre kerülnek

Fő (sikeres) szcenárió:
1. Minden node típus megjelenítésre kerül: <include>>Node típusok megjelenítése

Alternatív szcenáriók:

Használati eset: Node típusok megjelenítése
Elsődleges aktor: Felhasználó
Szereplők és érdekeik:
Felhasználó – adott node típus kerüjön megjelenítésere
Elő feltételek:
- Legyenek betöltött nodek a rendszerben

Sikeres végrehajtás (utófeltételek):
- A node típusok megjelenítésre kerültek a felhasználó számára

Fő (sikeres) szcenárió:
1. A rendszerben található nodek egyesével történő elemzése
2. Típusok eltárolása
3. Eltárolt típusok megjelenítése

Alternatív szcenáriók:

Használati eset: Node típus kijelölése
Elsődleges aktor: Felhasználó
Szereplők és érdekeik:
Felhasználó – egy node típus kijelölésre került későbbi műveletekhez
Elő feltételek:
- Létezzen node típus

Sikeres végrehajtás (utófeltételek):
- A kívánt node típus megjelöltként regisztrálva

Fő (sikeres) szcenárió:
1. A felhasználó kezdeményezi a node típus kijelölését
2. A rendszer megjelöli a node típust

Alternatív szcenáriók:

Használati eset: Node típus súgó megjelenítése
Elsődleges aktor: Felhasználó
Szereplők és érdekeik:
Felhasználó – az adott nodetípushoz tartozó leírás olvasása
Elő feltételek:
- Létezzen egy kijelölt nodetípus <<include>> Node típus kijelölése

Sikeres végrehajtás (utófeltételek):
- Súgó szöveg prezentálása a felhasználó számára

Fő (sikeres) szcenárió:
1. Súgó szöveg kikérése a node típustól
2. Súgó szöveg megjelenítése

Alternatív szcenáriók:
1a: nincsen deffiniált súgó szöveg:
1. üzenet küldése a felhasználó részére

Használati eset: Node hozzáadása
Elsődleges aktor: Felhasználó
Szereplők és érdekeik:
Felhasználó – egy új node példány készítése és hozzáadása a process chainhez
Elő feltételek:
- Lézetten egy Node típus kijelölve: <<include>>Node típus kijelölése
- Létezzen egy process chain

Sikeres végrehajtás (utófeltételek):
- A node példány elkészült és letárolásre került a process chainben

Fő (sikeres) szcenárió:
1. Adott node típus kérvényezése a NodeManagertől Node típus leíró alapján
2. A node példány tárolása process chainben

Alternatív szcenáriók:
1a: nem érvényes típus:
1.: a hozzáadási művelet megszakad

Használati eset: Node klónozása
Elsődleges aktor: Felhasználó
Szereplők és érdekeik:
Felhasználó – Egy node másolása
Elő feltételek:
- Létezzen egy Node példány az aktuális process chainben: <<include>> Node kijelölése

Sikeres végrehajtás (utófeltételek):

- Az új elem hozzáadásra került a process chainhez

Fő (sikeres) szcenárió:

1. A kijelölt node típus leírójával egy új node hozzáadásának kezdeményezése: <<include>>Node hozzáadása

Alternatív szcenáriók:

Használati eset: Node kijelölése
Elsődleges aktor: Felhasználó
Szereplők és érdekeik:
Felhasználó – Egy Node kijelölése további műveletekre
Elő feltételek:

- Létezzen egy Node példány az aktuális process chainben

Sikeres végrehajtás (utófeltételek):

- Az elem kijelöltként regisztrálásra került

Fő (sikeres) szcenárió:

1. A felhasználó kezdeményezi egy node példány kijelölését
2. A rendszer kijelöltként regisztrálj a a node példányt

Alternatív szcenáriók:

Használati eset: Node megjelenítése
Elsődleges aktor:
Szereplők és érdekeik:
Elő feltételek:

- Létezzen egy node példány

Sikeres végrehajtás (utófeltételek):

- A node példány megjelenítésre került

Fő (sikeres) szcenárió:

1. Az adott node összes slotját jelenítsük meg: <<include>>Node slot megjelenítése
2. Az adott node összes parameter-et jelenítsük meg: <<include>>Parameter megjelenítése

Alternatív szcenáriók:

Használati eset: Node slot megjelenítése
Elsődleges aktor: Felhasználó
Szereplők és érdekeik:
Felhasználó – egy node slot megjelenítése
Elő feltételek:

- Létezzen egy node slot

Sikeres végrehajtás (utófeltételek):

- A node slot megjelenítésre került a felhasználó számára

Fő (sikeres) szcenárió:

1. A nodeslot elemeinek kirajzoltatásra

Alternatív szcenáriók:

1. a: kirajzoltatás hibába ütközik
    1. megjelenítés megszakad

Használati eset: Parameter megjelenítése
Elsődleges aktor: Felhasználó
Szereplők és érdekeik:

Felhasználó – egy parameter megjelenítése

Elő feltételek:

- Létezzen egy parameter

Sikeres végrehajtás (utófeltételek):

- A parameter megjelenítésre került a felhasználó számára

Fő (sikeres) szcenárió:

1. A parameter elemeinek kirajzoltatásra

Alternatív szcenáriók:

1. a: kirajzoltatás hibába ütközik
    1. megjelenítés megszakad

Használati eset: Parameter szerkesztése
Elsődleges aktor: Felhasználó
Szereplők és érdekeik:

Felhasználó – egy parameter sikeres szerkesztése

Elő feltételek:

- Létezzen egy valid parameter és egy valid regisztrált gui elem amely alkalmas az adott parameter típus szerkeszétésre
- <<include>>Parametere Kijelölése

Sikeres végrehajtás (utófeltételek):

- Az adott paraméter szerkesztése sikeres volt az adatok mentése megtörént

Fő (sikeres) szcenárió:

1. A felhasználó kezdeményezik az adott parameter szerkeszétés
2. A rendszer kikeresi az adott parameterhez tartozó szerkesztő objektumot, amellyel megadja a lehetőséget a paraméter szerkesztésére
3. A felhasználó elvégzi a szerkeszétsi műveletet
4. A rendszer elmenti a változásokat

Alternatív szcenáriók:

2. a: Nincsen regisztrált szerkesztő az adott típushoz

1. a rendszer az adott parametert konvertálni próbálja egy olyan típusra amelyhez létezik regisztrált szerekesztő elem

2. ha talált ilyen elemet akkor ezzel folytatódik tovább a művelet: 3. pont

3. ha nem talált ilyen elemet akkor a felhasználót értesíti a hibáról

---

Használati eset: Slot Connection létrehozása
Elsődleges aktor: Felhasználó
Szereplők és érdekeik:

Felhasználó – kapcsolat létrehozása 2 slot között

Elő feltételek:

- létezzen 2 slot

Sikeres végrehajtás (utófeltételek):

- A két slot között létrejött a kapcsolat amelyet a process chain használhat

Fő (sikeres) szcenárió:

1. A felhasználó kijelöli az első slot-ot
2. A felhasználó kijelöli a második slot-ot
3. A felhasználó kezdeményezi a slotok összekapcsolását
4. A rendszer ellenőrzi a slotok irányát: csak KI és BE slot kapcsolható össze
5. A rendszer ellenőrzi a BE meneti slot foglaltságát
6. A rendszer ellenőrzi a slotok parametereinek típus azonosságát
7. A rendszer létrehozza a kapcsolatot a két slot között

Alternatív szcenáriók:

4.a: rossz a slotok iránya

1.: a kapcsolódás meghiúsul: a rendszer üzenetet küld a felhasználónak

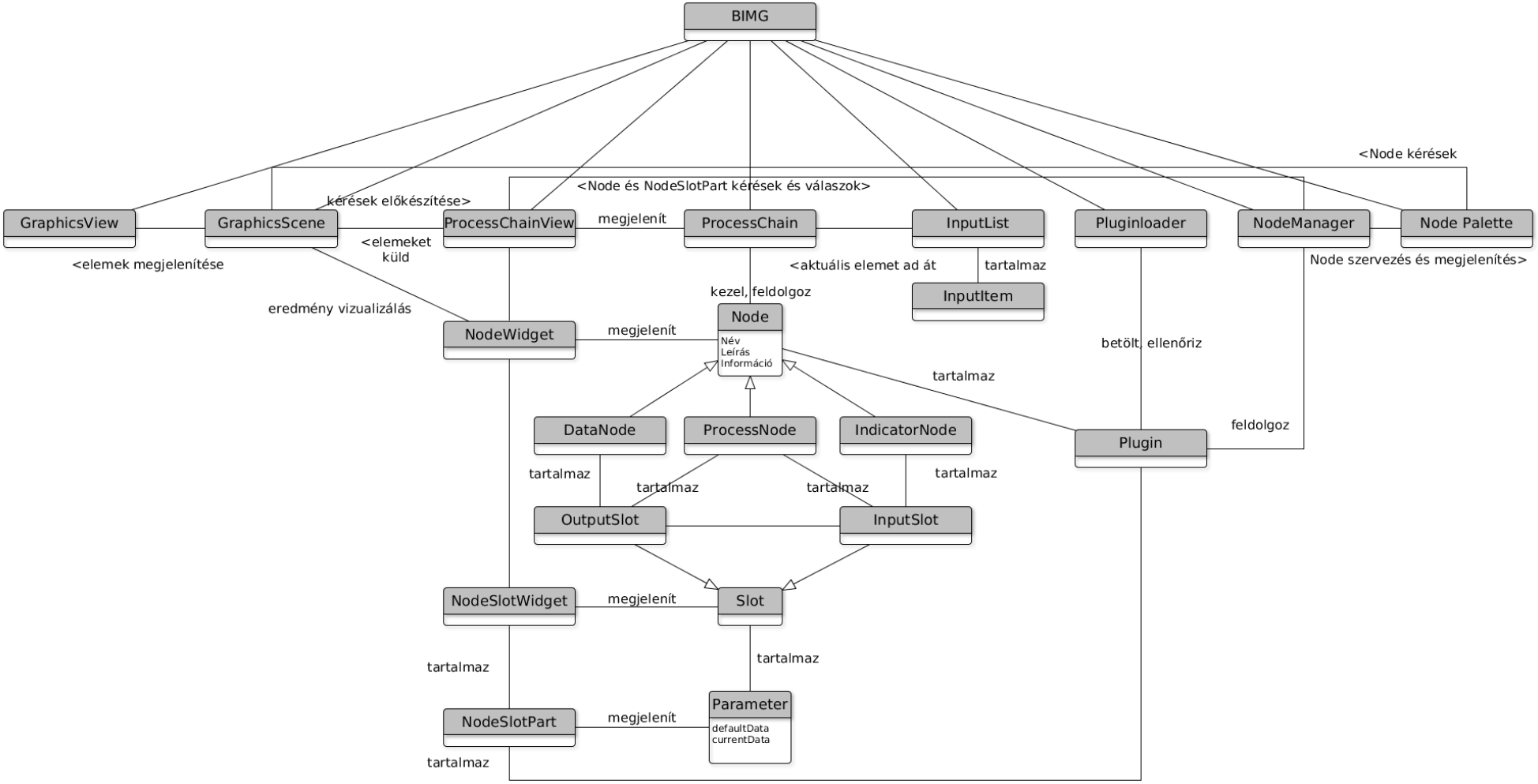
5.a: a bemeneti slot már rendelkezik kapcsolattal

1.: a kapcsolódás meghiúsul: a rendszer üzenetet küld a felhasználónak

6.a: a slotok parameterjei nem azonosak

1.: a rendszer parameter konverziót kísérel meg ha sikeres kapcsolat létrejön

10.3. Domain model



4. ábra. A domain model állapota az utolsó fejlesztési iteráció végén

10.4. Felhasználói útmutató

10.5. CD melléklet

A szakdolgozat CD mellékletének könyvtárszerkezete:

/VargaMarcell-DVLKHU-szakdolgozat.pdf

/szakdolgozat-forraskod

- /diagramok
- /szakdolgozat.tex

/rendszer-forraskod

- /src
  - /Szakdolgozat

/internetes-hivatkozasok

- /1_osszehasonlitas
- /2_kovetelmenyanalizis